Deep Natural Language Processing

Jason Weston
Facebook, NY.

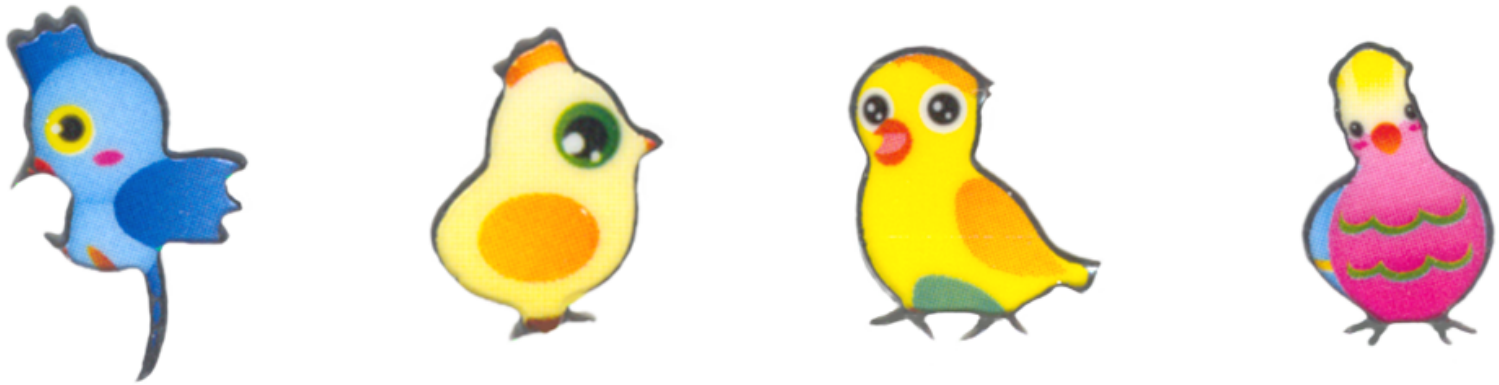
About me (Jason Weston)

2000 PhD from Royal Holloway, University of London
(advisors: Gammerman, Vovk, Vapnik).
2000-2002 Bioinformatics Startup (Biowulf).
2002-2003 Research Scientist at Max Planck Institute for Biological Cybernetics.
2003-2008 Research Scientist at NEC Labs America, Princeton.
2008-2014 Research Scientist at Google, NY.
2014+ Research Scientist at Facebook, NY.

What I've worked on:
- Kernel methods (multiclass SVM formulation, structured output learning, semi-supervised learning).
- Deep learning (Neural Net) algorithms for text, images.
- Machine Learning methods for Bioinformatics.
- Large scale learning for NLP, retrieval and recommendation.
- Google products: Web Search, Image Search, YouTube, Google Music, Google Products, Play Store, Now, Ads.

Part 1

Deep NLP Tagging



Based on:

Collobert, R., Weston, J., Bottou, L., Karlen, M., Kavukcuoglu, K., & Kuksa, P. (2011). Natural language processing (almost) from scratch. The Journal of Machine Learning Research, 12, 2493-2537.

NLP Tasks

Part-Of-Speech Tagging (POS): syntactic roles (noun, adverb...)

Chunking: syntactic constituents (noun phrase, verb phrase...)

Name Entity Recognition (NER): person/company/location...

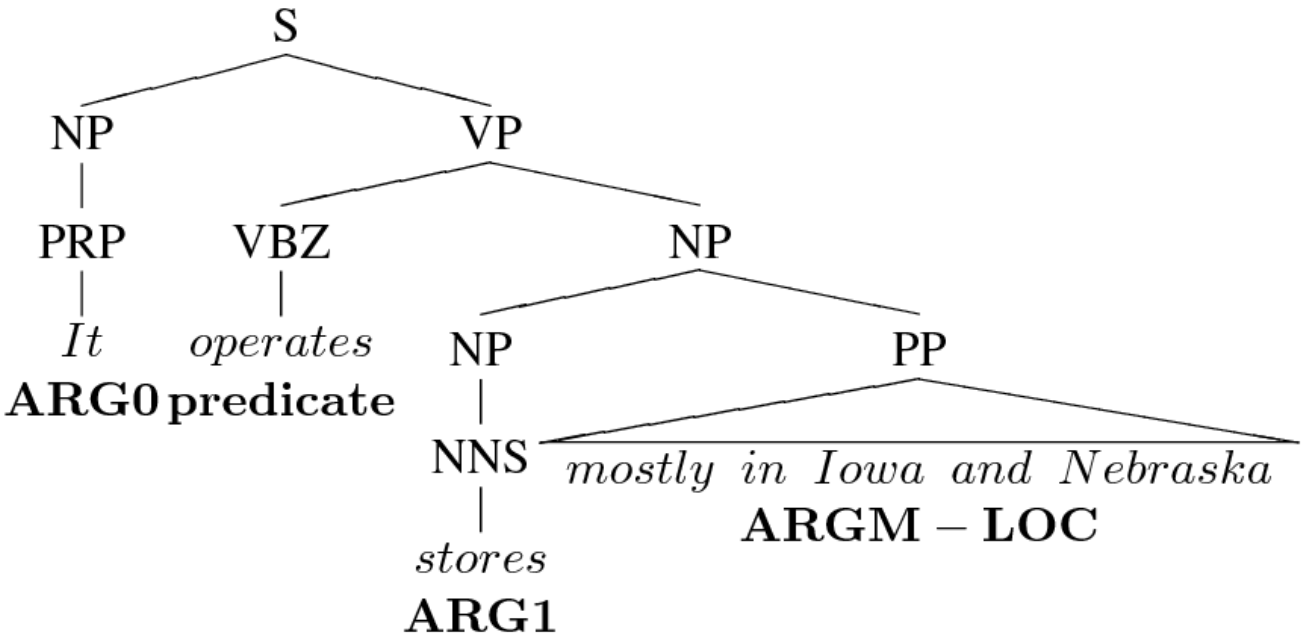
Semantic Role Labeling (SRL):

[John]$_{ARG0}$ [ate]$_{REL}$ [the apple]$_{ARG1}$ [in the garden]$_{ARGM-LOC}$

Complex Systems

- Two extreme choices to get a complex system
  - ⋆ Large Scale Engineering: design a lot of complex features, use a fast existing linear machine learning algorithm
  - ⋆ Large Scale Machine Learning: use simple features, design a complex model which will implicitly learn the right features

The Shallow System Way



- Extract **hand-made features** e.g. from the parse tree
- Disjoint: all tasks trained separately, Cascade features
- Feed these features to a shallow classifier like SVM

ASSERT: many hand built features for SRL

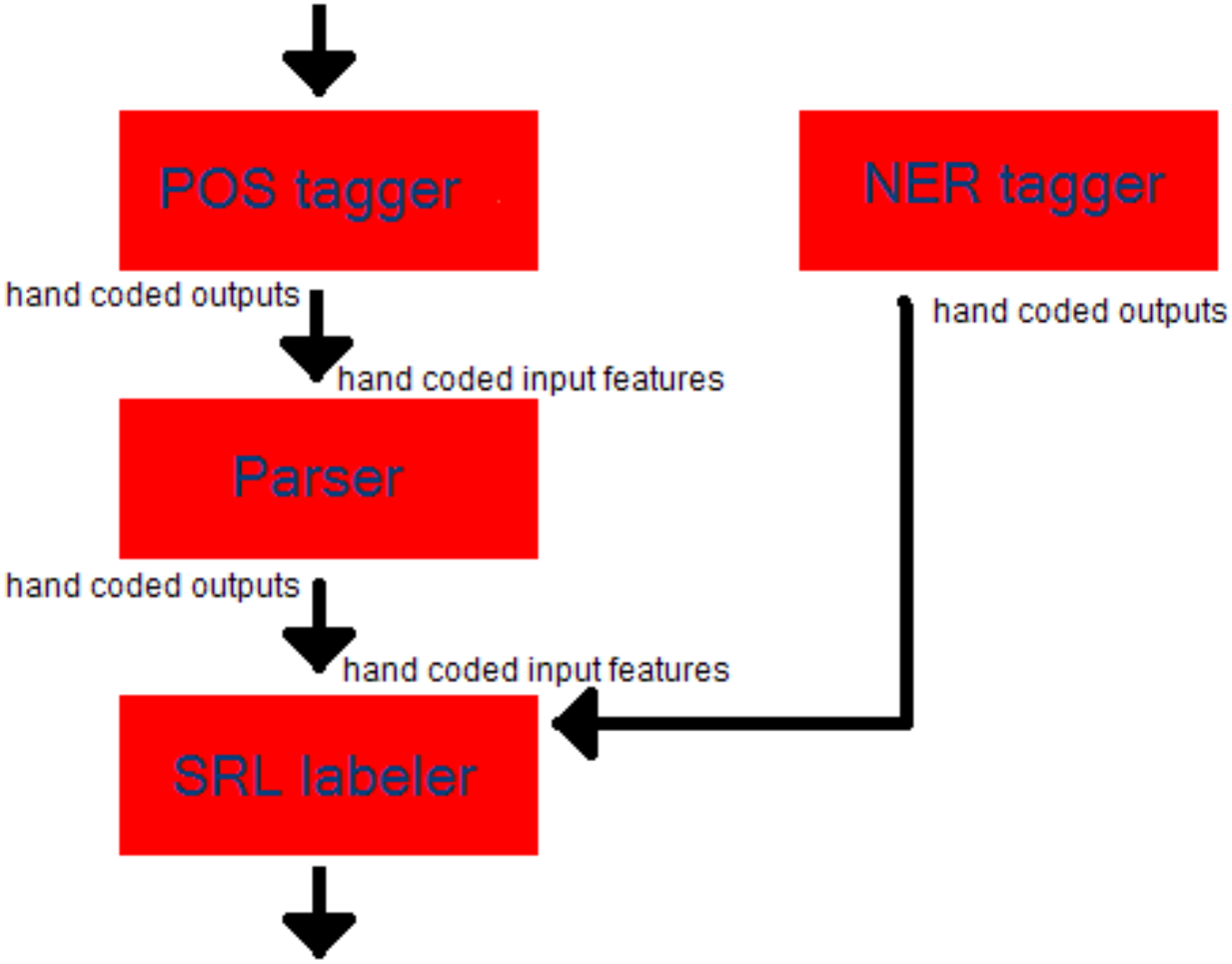
Problems:
1) Features rely on other solutions (parsing, named entity, word-sense)
2) Technology task-transfer is difficult

- Choose some **good hand-crafted features**

Predicate and POS tag of predicate	Voice: active or passive (hand-built rules)
Phrase type: adverbial phrase, prepositional phrase, ...	Governing category: Parent node's phrase type(s)
Head word and POS tag of the head word	Position: left or right of verb
Path: traversal from predicate to constituent	Predicted named entity class
Word-sense disambiguation of the verb	Verb clustering
Length of the target constituent (number of words)	NEG feature: whether the verb chunk has a "not"
Partial Path: lowest common ancestor in path	Head word replacement in prepopositional phrases
First and last words and POS in constituents	Ordinal position from predicate + constituent type
Constituent tree distance	Temporal cue words (hand-built rules)
Dynamic class context: previous node labels	Constituent relative features: phrase type
Constituent relative features: head word	Constituent relative features: head word POS
Constituent relative features: siblings	Number of pirates existing in the world...

- Feed them to a **shallow classifier** like SVM

The Suboptimal (?) Cascade



NLP: Large Scale Machine Learning

Goals

- Task-specific engineering limits NLP scope
- Can we find unified hidden representations?
- Can we build unified NLP architecture?

Means

- Start from scratch: forget (most of) NLP knowledge
- Compare against classical NLP benchmarks
- Our dogma: avoid task-specific engineering

NLP Benchmarks

- Datasets:
  - ⋆ POS, CHUNK, SRL: WSJ (≈ up to 1M labeled words)
  - ⋆ NER: Reuters (≈ 200K labeled words)

System	Accuracy
Shen, 2007	97.33%
Toutanova, 2003	**97.24%**
Gimenez, 2004	97.16%

(a) POS: As in (Toutanova, 2003)

System	F1
Shen, 2005	95.23%
Sha, 2003	**94.29%**
Kudoh, 2001	93.91%

(b) CHUNK: CoNLL 2000

System	F1
Ando, 2005	**89.31%**
Florian, 2003	88.76%
Kudoh, 2001	88.31%

(c) NER: CoNLL 2003

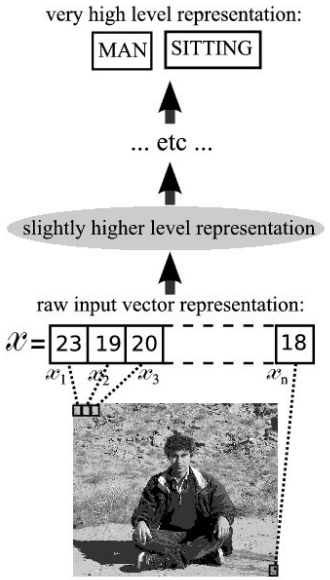
System	F1
Koomen, 2005	**77.92%**
Pradhan, 2005	77.30%
Haghighi, 2005	77.04%

(d) SRL: CoNLL 2005

- We chose as benchmark systems:
  - ⋆ Well-established systems
  - ⋆ Systems avoiding external labeled data
- Notes:
  - ⋆ Ando, 2005 uses external unlabeled data
  - ⋆ Koomen, 2005 uses 4 parse trees not provided by the challenge

The "Deep Learning" Way

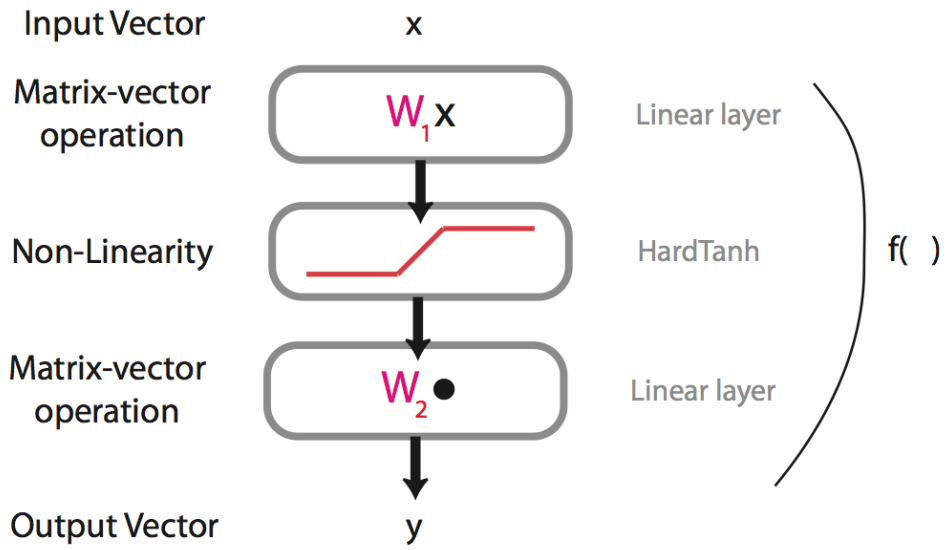
Deep approach proposes a radically different end-to-end approach:



- Avoid building a parse tree. Humans don't need this to talk.
- Try to avoid all hand-built features $\rightarrow$ monolithic systems.
- Humans implicitly learn these features. Neural networks can too. . . ?

Neural Networks

- Stack several layers together

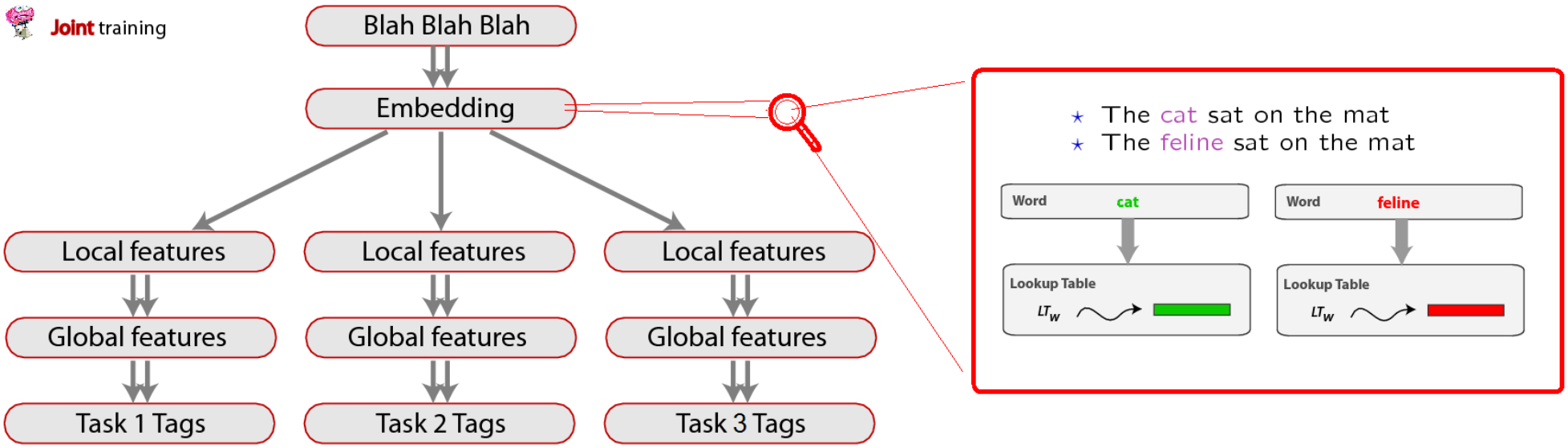


- Increasing level of abstraction at each layer
- Requires simpler features than "shallow" classifiers
- The "weights" W_i are trained by gradient descent
- How can we feed words?

The Big Picture

A unified architecture for all NLP (labeling) tasks:

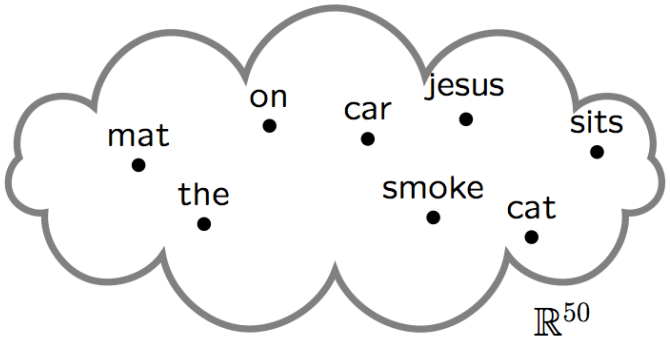
Sentence:	*Felix*	*sat*	*on*	*the*	*mat*	.
POS:	NNP	VBD	IN	DT	NN	.
CHUNK:	NP	VP	PP	NP	NP-I	.
NER:	PER	-	-	-	-	-
SRL:	ARG1	REL	ARG2	ARG2-I	ARG2-I	-



Words into Vectors

Idea

- Words are embed in a vector space



- Embeddings are trained

Implementation

- A word w is an index in a dictionary $\mathcal{D} \in \mathbb{N}$
- Use a lookup-table ($W \sim$ feature size $\times$ dictionary size)

$$LT_W(w) = W_{\bullet\, w}$$

Remarks

- Applicable to any discrete feature (words, caps, stems...)
- See (Bengio et al, 2001)

The Lookup Tables

Each word/element in dictionary maps to a vector in $\mathbb{R}^d$.

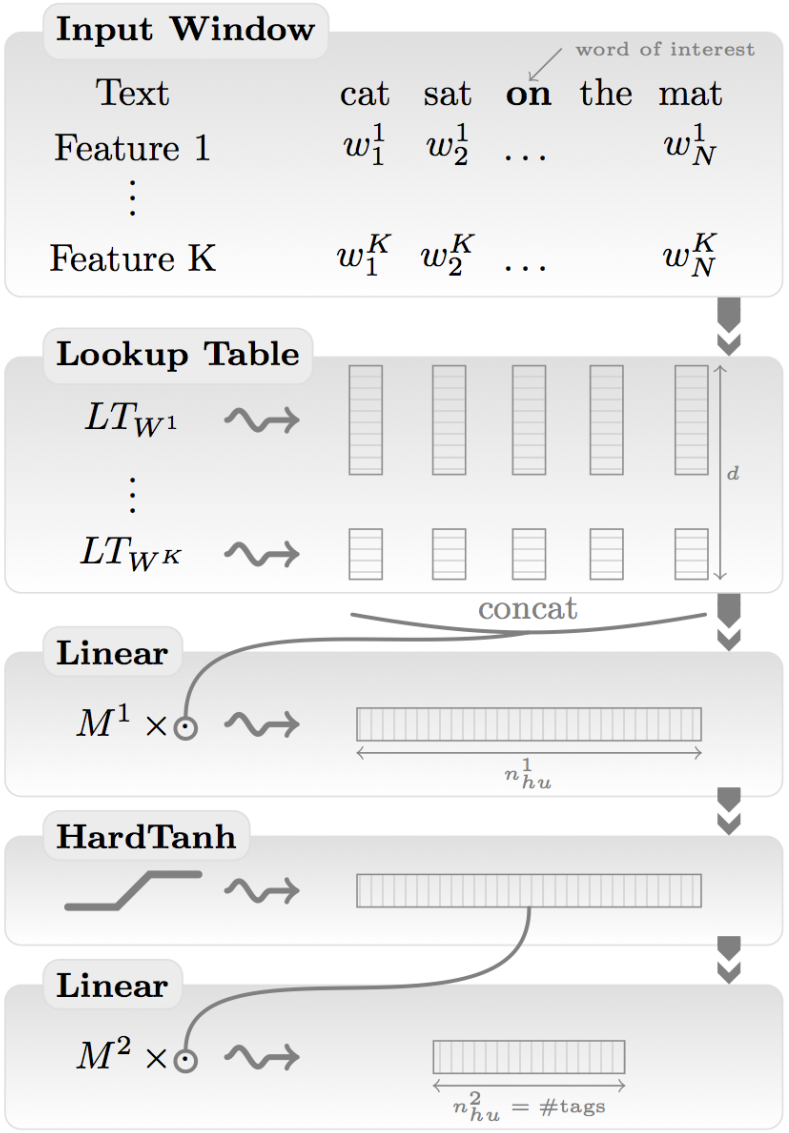
- We learn these vectors.
- LookupTable: input of i^{th} word is

$$x = (0, 0, \ldots, 1, 0, \ldots, 0) \quad 1 \text{ at position } i$$

In the original space words are orthogonal.
cat = (0,0,0,0,0,0,0,0,0,0,1,0,0,0,0, . . .)
kitten = (0,0,1,0,0,0,0,0,0,0,0,0,0,0,0, . . .)

To get the $\mathbb{R}^d$ embedding vector for the word we multiply Wx where W is a $d \times N$ vector with N words in the dictionary.

Window Approach

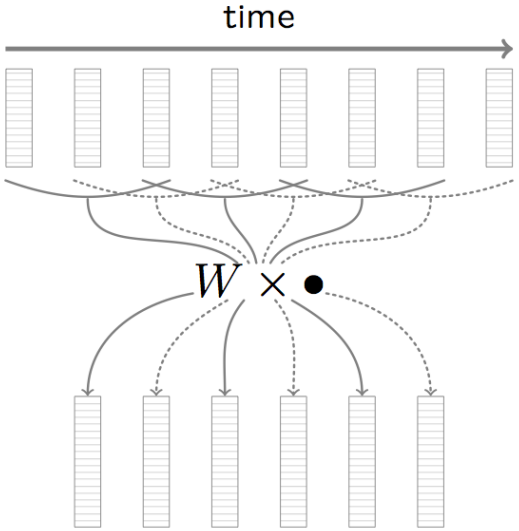


- Tags one word at the time
- Feed a fixed-size window of text around each word to tag
- Works fine for most tasks
- How do deal with long-range dependencies?

  E.g. in SRL, the verb of interest might be outside the window!

Sentence Approach (1/2)

- Feed the whole sentence to the network
- Tag one word at the time: add extra position features
- Convolutions to handle variable-length inputs

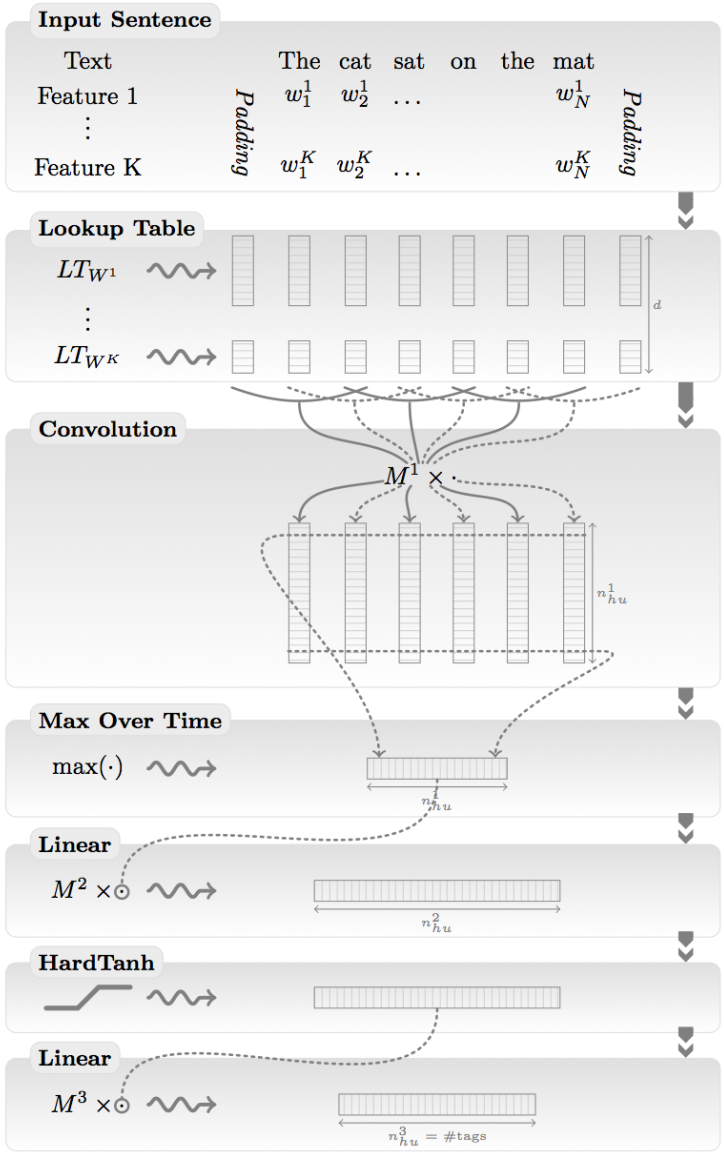


See (Bottou, 1989) or (LeCun, 1989).

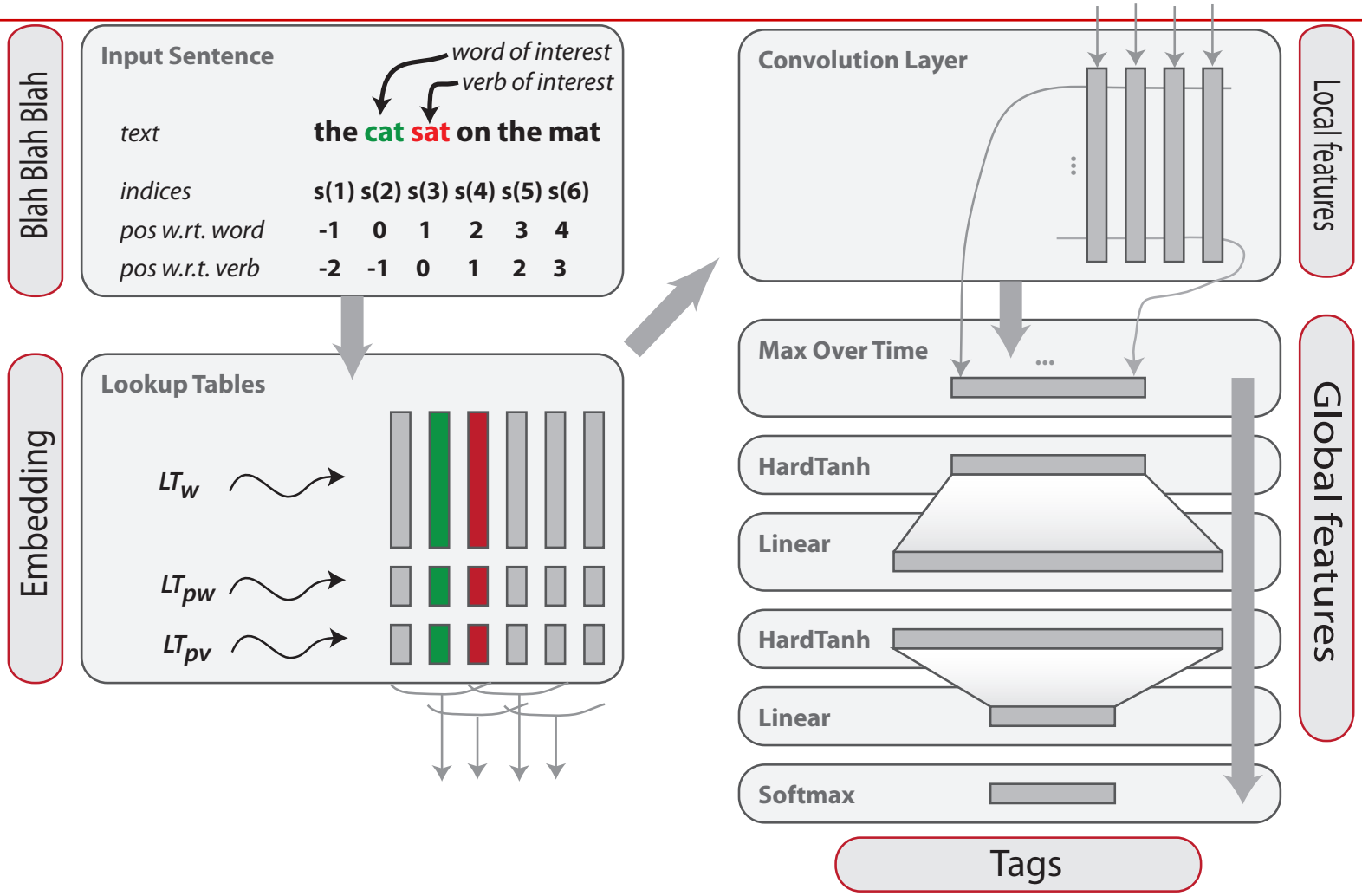
- Produces local features with higher level of abstraction
- Max over time to capture most relevant features

Outputs a fixed-sized feature vector

Sentence Approach (2/2)

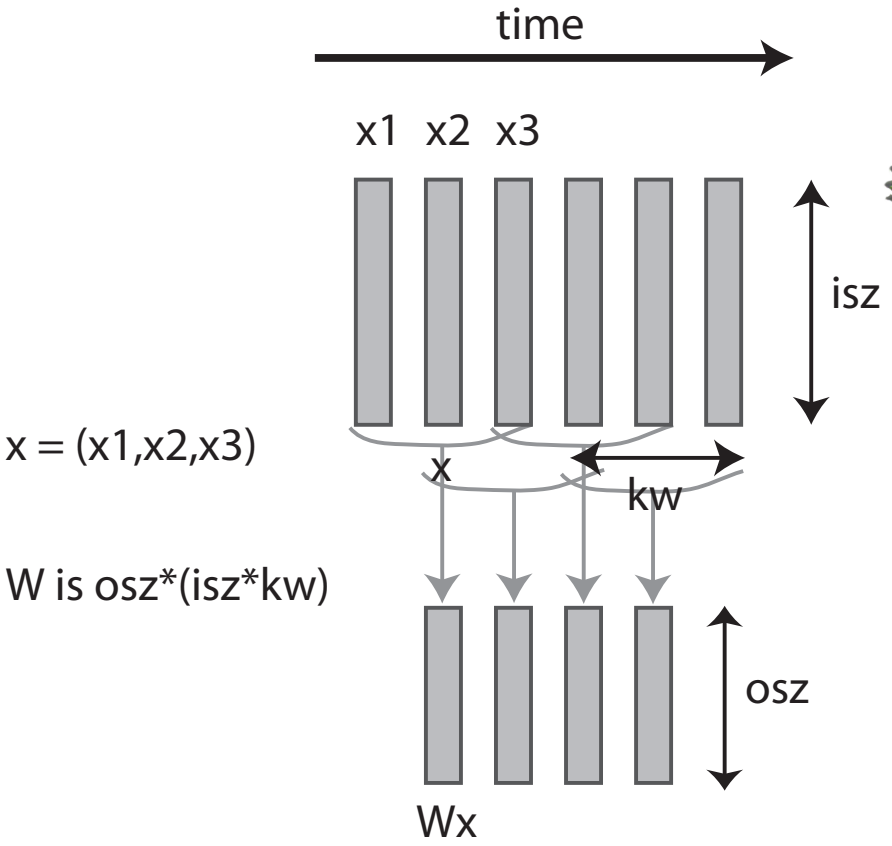


Deep SRL



This is the network for a single window. We train/test predicting the entire sentence of tags ("structured outputs") using viterbi approach, similar to other NLP methods.

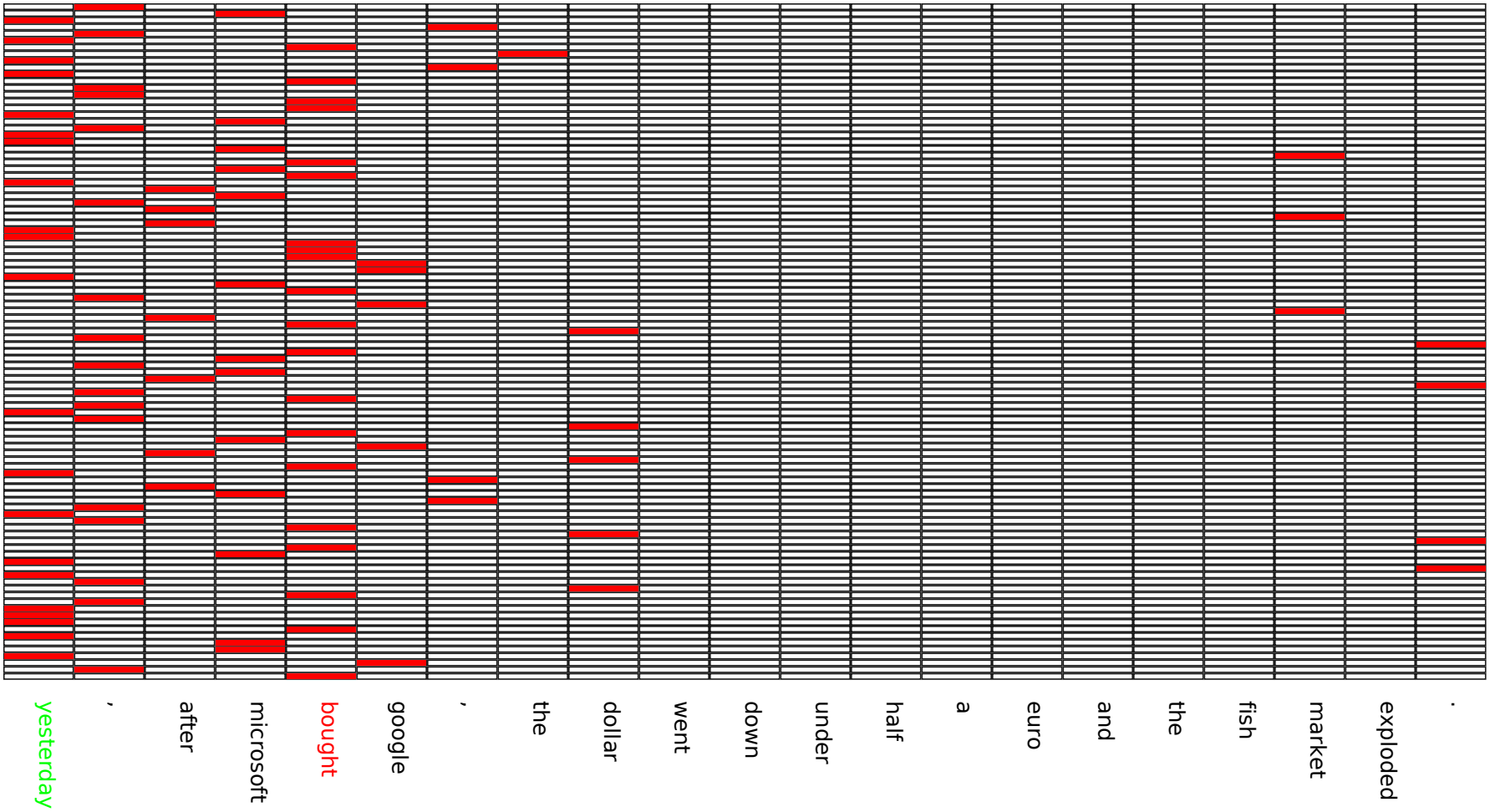
Convolutions



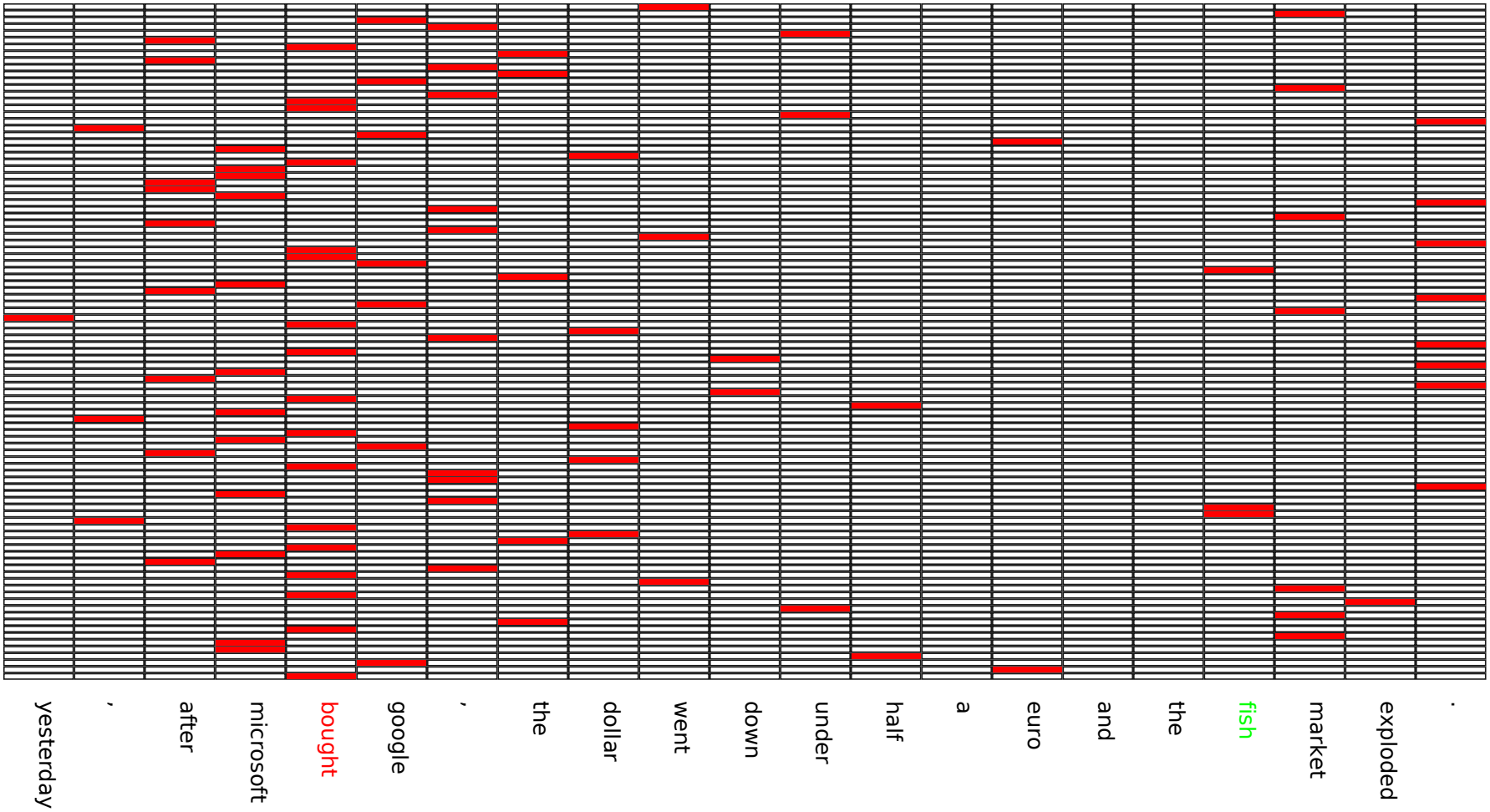
Extract local features – share weights through time/space

Used with success in image (Le Cun, 1989) and speech (Bottou & Haffner, 1989)

Removing The Time Dimension (1/2)



Removing The Time Dimension (2/2)



Word Tag Likelihood (WTL)

- The network has one output $f(\boldsymbol{x}, i, \boldsymbol{\theta})$ per tag i
- Interpreted as a probability with a softmax over all tags

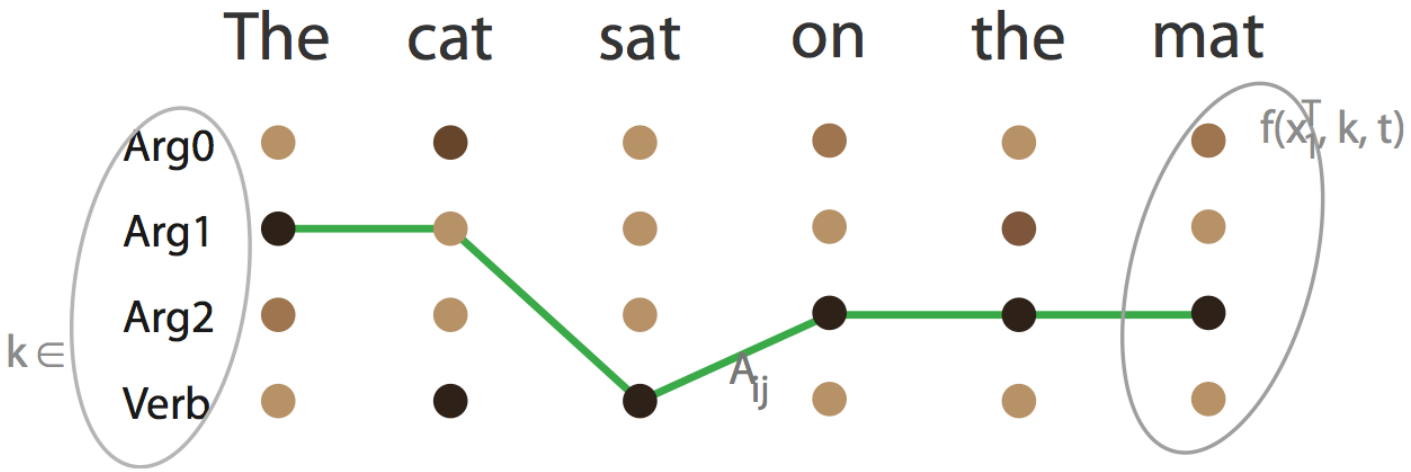
$$p(i \mid \boldsymbol{x}, \boldsymbol{\theta}) = \frac{e^{f(\boldsymbol{x}, i, \boldsymbol{\theta})}}{\sum_j e^{f(\boldsymbol{x}, j, \boldsymbol{\theta})}}$$

.. we can train directly for that (word tag likelihood) or we could train in a structured way by predicting the entire sentence's tags.

That should be useful because tags are not independent.

Sentence Tag Likelihood (STL)

- The network score for tag k at the t^{th} word is $f([\boldsymbol{x}]_1^T, k, t, \boldsymbol{\theta})$
- A_{kl} transition score to jump from tag k to tag l



- Sentence score for a tag path $[i]_1^T$

$$s([\boldsymbol{x}]_1^T, [i]_1^T, \tilde{\boldsymbol{\theta}}) = \sum_{t=1}^{T} \left(A_{[i]_{t-1}[i]_t} + f([\boldsymbol{x}]_1^T, [i]_t, t, \boldsymbol{\theta}) \right)$$

Supervised Benchmark Results

- Network architectures:
  - ⋆ Window (5) approach for POS, CHUNK & NER (300HU)
  - ⋆ Convolutional (3) for SRL (300+500HU)
  - ⋆ Word Tag Likelihood (WTL) and Sentence Tag Likelihood (STL)

- Network features: lower case words (size 50), capital letters (size 5)
  dictionary size 100,000 words

Approach	**POS** (PWA)	**Chunking** (F1)	**NER** (F1)	**SRL** (F1)
Benchmark Systems	**97.24**	**94.29**	**89.31**	**77.92**
NN+WTL	96.31	89.13	79.53	55.40
NN+STL	96.37	90.33	81.47	*70.99*

- STL helps, but... fair performance.
- Capacity mainly in words features... are we training it right?

Supervised Word Embeddings

- Sentences with similar words should be tagged in the same way:
  - ⋆ The cat sat on the mat
  - ⋆ The feline sat on the mat

france	jesus	xbox	reddish	scratched	megabits
454	1973	6909	11724	29869	87025
persuade	thickets	decadent	widescreen	odd	ppa
faw	savary	divo	antica	anchieta	uddin
blackstock	sympathetic	verus	shabby	emigration	biologically
giorgi	jfk	oxide	awe	marking	kayak
shaheed	khwarazm	urbina	thud	heuer	mclarens
rumelia	stationery	epos	occupant	sambhaji	gladwin
planum	ilias	eglinton	revised	worshippers	centrally
goa'uld	gsNUMBER	edging	leavened	ritsuko	indonesia
collation	operator	frg	pandionidae	lifeless	moneo
bacha	w.j.	namsos	shirt	mahan	nilgiris

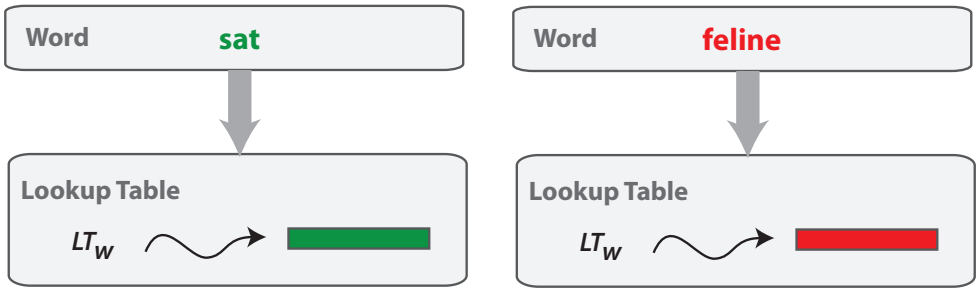
- About 1M of words in WSJ
- 15% of most frequent words in the dictionary are seen 90% of the time
- Cannot expect words to be trained properly!

Improving Word Embedding

Rare words are not trained properly

Sentences with similar words should be tagged in the same way:

- ⋆ The cat sat on the mat
- ⋆ The feline sat on the mat



Only 1M WSJ not enough – let's use lots of unsupervised data!

Semi-supervised: MTL with Unlabeled Text

- Language Model: "*is a sentence actually english or not?*"
  Implicitly captures: $\star$ syntax $\star$ semantics
- Bengio & Ducharme (2001) Probability of next word given previous words. Overcomplicated – we do not need probabilities here
- English sentence windows: Wikipedia ($\sim 631M$ words)
  Non-english sentence windows: middle word randomly replaced
  the champion federer wins wimbledon
  vs. the champion saucepan wins wimbledon
- Multi-class margin cost:

$$\sum_{s \in \mathcal{S}} \sum_{w \in \mathcal{D}} \max(0,\, 1 - f(s,\, w_s^\star) + f(s,\, w))$$

$\mathcal{S}$: sentence windows $\mathcal{D}$: dictionary
$w_s^\star$: true middle word in s
$f(s, w)$: network score for sentence s and middle word w

Language Model: Embedding

Nearest neighbors in 100-dim. embedding space:

france	jesus	xbox	reddish	scratched
454	1973	6909	11724	29869
spain	christ	playstation	yellowish	smashed
italy	god	dreamcast	greenish	ripped
russia	resurrection	psNUMBER	brownish	brushed
poland	prayer	snes	bluish	hurled
england	yahweh	wii	creamy	grabbed
denmark	josephus	nes	whitish	tossed
germany	moses	nintendo	blackish	squeezed
portugal	sin	gamecube	silvery	blasted
sweden	heaven	psp	greyish	tangled
austria	salvation	amiga	paler	slashed

(Even fairly rare words are embedded well.)

Results

Algorithm	POS (PWA)	CHUNK (F1)	NER (F1)	SRL (F1)
Baselines	97.24 [Toutanova '03]	94.29 [Sha '03]	89.31 [Ando '05]	77.92 [Koomen '05]
NN + WTL	96.31	89.13	79.53	55.40
NN + STL	96.37	90.33	81.47	70.99
NN + LM + STL	97.22	94.10	88.67	74.15
NN + . . . + tricks	97.29 [+suffix]	94.32 [+POS]	89.95 [+gazetteer]	76.03 [+Parse Trees]

NOTES:

– Didn't compare to benchmarks that used external labeled data.

– [Ando '05] uses external unlabeled data.

– [Koomen '05] uses 4 parse trees not provided by the challenge. Using only 1 tree it gets 74.76.

Software

Code for tagging with POS, NER, CHUNK, SRL + parse trees:
`http://ml.nec-labs.com/senna/`

System	RAM (Mb)	Time (s)
Toutanova, 2003	1100	1065
Shen, 2007	2200	833
SENNA	32	4

(a) POS

System	RAM (Mb)	Time (s)
Koomen, 2005	3400	6253
SENNA	124	52

(b) SRL

See also Torch: `http://www.torch.ch`

Conclusion: Deep Learning for NLP tagging

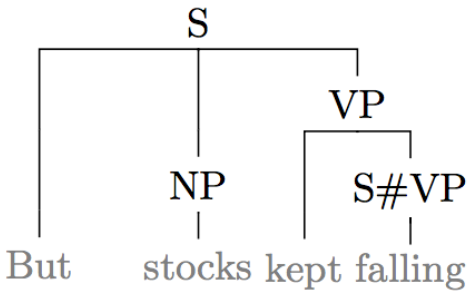
Generic end-to-end deep learning system for NLP tasks

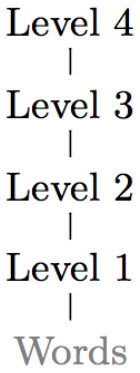
Previous common belief in NLP: engineering syntactic features necessary for semantic tasks.

One can do well by engineering an algorithm rather than features.

Attitude is changing in recent years...

Other Deep NLP work: Parsing





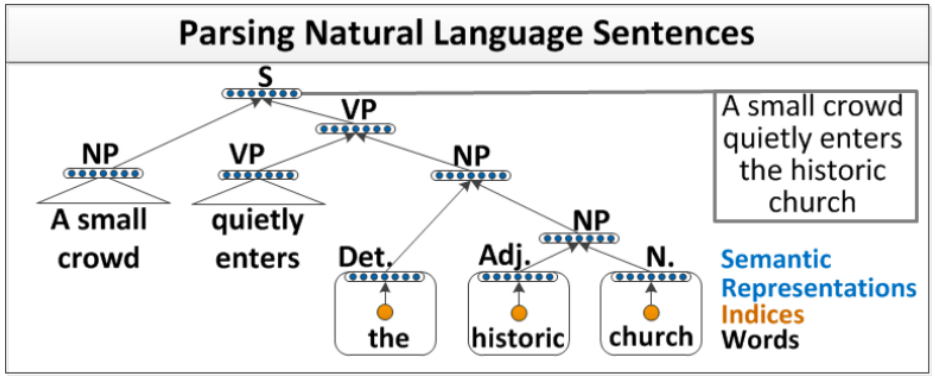
Level 4	B-S	I-S	I-S	E-S
Level 3	O	O	B-VP	E-VP
Level 2	O	O	O	S-S
Level 1	O	S-NP	O	S-VP
Words	But	stocks	kept	falling

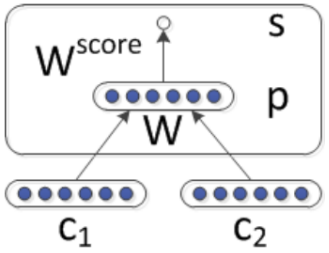
Figure 2: The parse tree shown in Figure 1a, rewritten as four levels of tagging tasks.

Solve as a (structured) tagging task, similar to methods before.

R. Collobert. Deep Learning for Efficient Discriminative Parsing. In AISTATS, 2011.

Other Deep NLP work: Recursive NNs for Parsing, Sentiment, ...



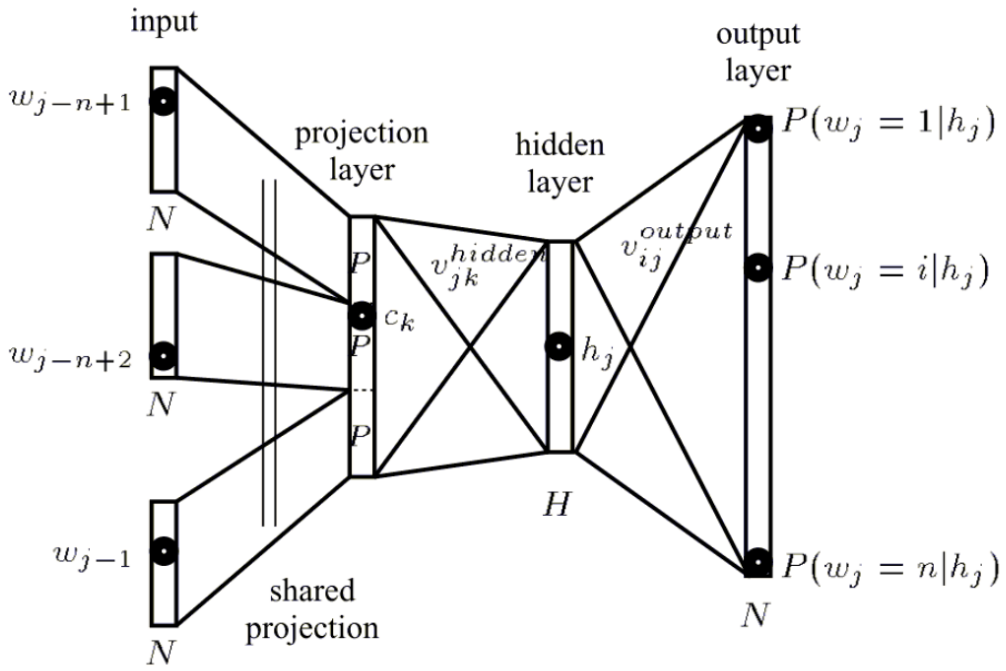


$$s = W^{score}p \quad (9)$$
$$p = f(W[c_1; c_2] + b)$$

Socher, Richard, et al. "Parsing natural scenes and natural language with recursive neural networks." Proceedings of the 28th International Conference on Machine Learning (ICML-11). 2011.

Socher, Richard, et al. "Recursive deep models for semantic compositionality over a sentiment treebank." Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP). 2013.

Other Deep NLP work: Language Models



Bengio, Y., Schwenk, H., Sencal, J. S., Morin, F., & Gauvain, J. L. (2006). Neural probabilistic language models. In Innovations in Machine Learning (pp. 137-186). Springer Berlin Heidelberg.

Other Deep NLP work: RNN LMs

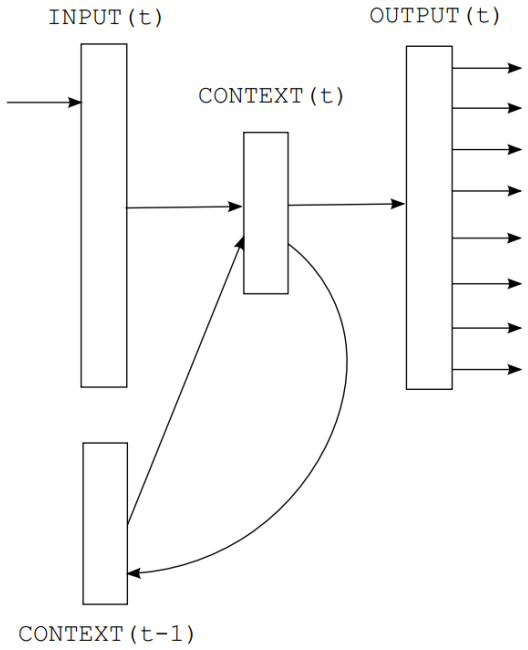
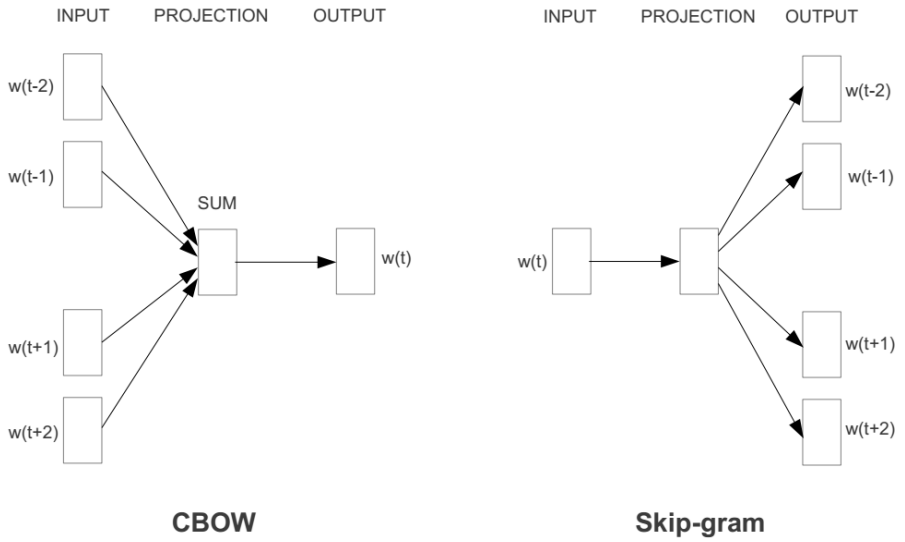


Figure: *Recurrent neural network based LM*

Recurrent neural network based language model. T Mikolov, M Karafit, L Burget, J Cernock, S Khudanpur INTERSPEECH, 1045-1048.

Other Deep NLP work: word2vec LMs



Czech + currency	Vietnam + capital	German + airlines	Russian + river	French + actress
koruna	Hanoi	airline Lufthansa	Moscow	Juliette Binoche
Check crown	Ho Chi Minh City	carrier Lufthansa	Volga River	Vanessa Paradis
Polish zolty	Viet Nam	flag carrier Lufthansa	upriver	Charlotte Gainsbourg
CTK	Vietnamese	Lufthansa	Russia	Cecile De

Table 5: Vector compositionality using element-wise addition. Four closest tokens to the sum of two vectors are shown, using the best Skip-gram model.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. NIPS, 2013.

Work on using text embedding for semantics:

More work that I have done (with co-authors):

A. Bordes, X. Glorot, J. Weston and Y. Bengio.
Joint Learning of Words and Meaning Representations for Open-Text Semantic Parsing. AISTATS 2012.

A. Bordes, J. Weston, R. Collobert and Y. Bengio.
Learning Structured Embeddings of Knowledge Bases. AAAI 2011.

A. Bordes, N. Usunier, R. Collobert, J. Weston.
Towards Understanding Situated Natural Language. AISTATS, 2010.

They use similar tools to incorporate world knowledge into text understanding by embedding concepts and words in a joint space.

Work on using text embedding for IR:

More work that I have done (with co-authors):

B. Bai, J. Weston, D. Grangier, R. Collobert, K. Sadamasa, Y. Qi, C. Cortes, M. Mohri. Polynomial Semantic Indexing. NIPS, 2009.

J. Weston, S. Bengio, N. Usunier. Large Scale Image Annotation: Learning to Rank with Joint Word-Image Embeddings. ECML PKDD 2010 special issue of the Machine Learning journal.

A. Lucchi and J. Weston. Joint Image and Word Sense Discrimination For Image Retrieval. ECCV 2012.

They use similar tools to perform retrieval, the latter two using embeddings of words *and* images in a joint space.

Cheers!

.....from Facebook Labs!